EXHIBIT A

LISTING OF ALL CLAIMS AND AMENDMENTS

(06-10-2005)

Amendments to the Specification

Please amend the first paragraph on page 1 as follows:

This application claims priority of U.S. Provisional Patent Application Serial No. 60/260,~~261~~061, which was filed on January 4, 2001, the specification of which is incorporated herein by reference.

Please amend the second paragraph on page 4 as follows:

FIG. 1 is a module interaction map depicting the interaction of the primary modules of one ~~exemplary~~example server system of the present invention;

Please amend the third, fourth, and sixth paragraphs on page 5 as follows:

FIG. 15 is a scenario map depicting one ~~exemplary~~example initial connection process implemented by the server system of FIG. 1;

FIG. 16 is a scenario map depicting one ~~exemplary~~example method call process implemented by the server system of FIG. 1;

FIG. 18 is a scenario map depicting another ~~exemplary~~example method call process implemented by the server system of FIG. 1;

Please amend the first, second and third paragraphs on page 6 as follows:

The present invention may be embodied as a motion control server system comprising a number of modules. In the following discussion, the overall environment in which the present invention is typically used will first be described. Following that will be a detailed discussion of the interaction of the various modules that form one ~~exemplary~~example embodiment of the present invention. The ~~exemplary~~example

embodiment operates in a number of scenarios, and a number of these scenarios will then be described. Finally, certain components of the ~~exemplary~~example motion control server system, and typical use scenarios for these components, will be described in further detail.

Referring initially to ~~FIGS~~FIG. 1A of the drawing, depicted at 20a therein is a motion control server system constructed in accordance with, and embodying, the principles of the present invention. The ~~exemplary~~example motion control server system 20a is configured to transmit motion control data between a data source 22 and a motion control system 24 through a communications system 26. The ~~motion control data is represented by~~ data source 22 comprises or is formed at least in part by (see, for example FIG. 2) an application program 28 comprising methods, function calls, and/or data.

The ~~exemplary~~example motion control server system 20a comprises a service request format module 30 and a data format module 32. The service request format module 30 converts service requests (methods and/or function calls) of the application program 28 between a network service request format and a native service request format defined by the motion control system 24. The data format module 32 converts data sets transferred between the source 22 and the motion control system 24 between a network data format and a native data format defined by the motion control system 24.

On page 7, please amend the first paragraph, the title at line 9, and the second and third paragraphs as follows:

FIGS. 1B and 1C indicate that some benefits of the present invention may be obtained by using either one of the service request format module 30 and the data format module 32. Depicted in FIG. 1B is an alternative ~~exemplary~~example motion control server system 20b that employs only the data format module 32, while FIG. 1C depicts yet another ~~exemplary~~example motion control server system 20c employing only the service request format module 30.

II. ~~EXEMPLARY~~EXAMPLE MOTION CONTROL SERVER SYSTEM

Referring now to FIG. 1, depicted at 20 therein is one preferred embodiment of a motion control server system of the present invention. The motion control server system 20 will be described herein in the context of ~~one exemplary~~a particular data source 22, motion control system 24, and communications system 26. However, the present invention may be embodied in forms appropriate for other data sources, motion control systems, and communications systems. In addition, the preferred motion control server system 20 comprises a number of optional modules that are not necessary to carry out the principles of the present invention in a basic form.

Two sets of terminology will be used with reference to the motion control server system 20. The first set is generic and is applicable to any environment in which a motion control server system of the present invention may be used. The second is specific to the ~~exemplary~~example motion control server system 20 and the data source 22, motion control system 24, and communications system 26 in connection with which the server system 20 is used. In the following discussion, the major elements will be initially identified using the generic terminology, with the specific terminology being identified in parenthesis. After this initial introduction, both sets of terminology will be used interchangeably.

On page 8, please amend the first and second paragraphs, and line 19 as follows:

The ~~exemplary~~example motion control server system (XMC Internet Connect system) 20 comprises both the service request format module (XMC SOAP Engine) 30 and data format module (XMC XML Engine) 32. In addition, the ~~exemplary~~example server system 20 comprises two optional modules: a data caching module (XMC SQL Store) 40 and a method discovery module (XMC DynaDiscovery) 42. These components allow virtually any client application 28 to utilize the underlying motion control system 24 regardless of the client application's location, underlying hardware platform, or underlying software operating system.

These modules 30, 32, 40, and 42 are optimized to connect the data source (client machine or device) 22 to a motion services module (XMC Motion Services) 50 forming a part of the motion control system 24 over the communications system (Internet) 26. The XMC Motion Services module 50 is a hardware independent connection to the underlying motion control hardware system (not shown). The ~~exemplary~~example XMC Motion Services module 50 is described in detail in one or more of the following U.S. Patents 5,691,897, 5,867,385, and 6,209,037 B1 and will not be described herein in further detail.

The ~~exemplary~~example XMC SOAP Engine module 30 is based on an industry

Please amend the second full paragraph (beginning at line 10), and the third and fourth paragraph on page 9 as follows:

The XMC SQL Store module 40 is used to cache data queried from the XMC XML Engine 32 (or directly from the native XMC Motion Services module 50). The ~~exemplary~~example XMC SQL Store module 40 caches data in database module 44 (SQL database or other database such as Microsoft Access or Oracle, etc).

The XMC DynaDiscovery module 42 is used to 'discover' the services supported by both the XMC XML Engine 32 and native XMC Motion Service module 50. The ~~exemplary~~example method discovery module 42 is based on the industry standard DISCO (Discovery of Web Services) protocol.

As noted in Figure 1 above, there are also several other modules that optional~~optionally~~ may be used with or incorporated into the XMC Internet Connection server system 20. In particular, the server system 20 uses the motion services (XMC Motion Services) module 50, motion drivers (XMC Driver) 52, a process control (XMC OPC) module 54, a packet processing (ROPE) module 56, and a data management (Biztalk Server system 2000) module 58.

Please amend the second full paragraph (beginning at line 9), and the third, fourth and fifth paragraph on page 10 as follows:

The ~~exemplary~~example process control module 54 is a standard OPC (OLE for Process Control) server used to query and set data sets using the OPC protocols as defined by the OLE for Process Control Foundation.

The ~~exemplary~~example packet processing module 56 is a DLL module released by Microsoft and referred to as ROPE (Remote Object Proxy Engine). The ROPE module is specifically designed to build and parse SOAP data packets.

The ~~exemplary~~example data management module 58 is or may be the Microsoft BizTalk 2000 server. The Biztalk 2000 server is used to map data between XML Schemas, set up data agreements between companies, and manage data connections between organizations.

FIG. 1 further illustrates that the ~~exemplary~~example server system 20 employs a number of 'schemas' that are passed between modules. A 'schema' is a data format specification for XML data. Each schema determines how data following the protocol of the schema is organized. The ~~exemplary~~example server system 20 makes use of the following schemas: motion control (XMC) schemas 60, process control (OPC) schemas 62, database management (SQL) schemas 64, and third party schemas 66 such as the OMAC schema.

Please amend the first and second paragraphs under the heading *A. Service Discovery* on page 11 as follows:

Before a web service can be used, the services that service offers are determined or "discovered". Before discovering what a single web service can do, the web server is queried to determine what the web services that it offers. In the ~~exemplary~~example server system 20, the optional method discovery module 42 is used to discover the services available from the motion services module 50 using one or more of a plurality of protocols such as the Dynamic Web Discovery (DISCO) protocol, SNMP, LDAP, and the like. The ~~exemplary~~example XMC DynaDiscovery module 42

uses the DISCO protocol because the DISCO protocol is based on XML, which allows a very thin client to use the discovery service.

FIG. 2 of the drawing illustrates the steps that occur when the ~~exemplary~~example server system 20 uses the method discovery module 42 to discover the services available from the motion services module 50. First, the client application (or machine or device) 28, queries the motion control server system 20 for the services provided. This request may go through the BizTalk Server 58 or directly to the SOAP enabled server module 30.

Please amend the third full paragraph (beginning at line 13) on page 12 as follows:

Upon receiving the request, the XMC DynaDiscovery module 42 queries all modules that it 'knows about'. Such modules typically include or define type libraries (TLB) 72 that define the offered services. The ~~exemplary~~example module 42 thus examines the Type Libraries 72 to 'discover' the services that they offer. Upon discovering the available services, the DynaDiscovery module 42 dynamically builds an SDL (Services Description Language) data set and returns it to the requesting SOAP server 30. When dynamic discovery is not used, the SDL file is a static file that resides on the SOAP enabled server.

Please amend the first full paragraph (beginning at line 3) on page 16 as follows:

When requested, the ~~exemplary~~example XMC SOAP Engine uses the ROPE module 56 to parse the request and determine what service operation is to be performed. As discussed above, the use of the ROPE module 56 is not required but is preferred.

Please amend the first paragraph and line 27 on page 17 as follows:

The XMC XML Engine 32 is designed to be a 'middle-ware' component that translates data between a native system and XML. In the ~~exemplary~~example system 20, this translation is bi-directional. Translations from the XML data format to the data format of the native motion services module 50 are used to change configuration data and properties and to cause actions. Translations from the native data format of the motion services module 50 to XML data format are used when querying configuration data or properties.

FIG. 6 illustrates that the ~~exemplary~~example XMC XML Engine module 32

Please amend the first full paragraph (beginning at line 3) on page 18 as follows:

Referring now to FIG. 7, depicted at 80 therein is an interface map for the XMC XML Engine module 32. The ~~exemplary~~example XMC XML Engine module 32 implemented as a COM component that houses several objects. Each object exposes one or more OLE interfaces.

Please amend the last line (line 33) on page 28 as follows:

Referring initially to FIG. 13, depicted therein is the basic HTML Soap request as implemented using the data server system 20 of the present invention. To operate over a communications network 24 such as the Internet, the data server system 20 must be capable of receiving Internet/Web requests. FIG. 1 illustrates this capability by an internet information application programming interface (IIAPI) 74. FIG. 13 illustrates that the interface 74 is defined by an information server module 76; in the ~~exemplary~~example system 20, the information server module 76 is formed by a Microsoft Internet Information Server (IIS) based server installed with the XMC SOAP Engine 30.

Please amend the third full paragraph on page 34 (beginning at line 10) as follows:

The XMC SOAP Engine 30 builds on SOAP technology to obtain the data server system 20 that is enabled for motion-based application. In particular, the ~~exemplary~~example XMC SOAP Engine 30 extends information server module 76 to support SOAP requests and routes each request appropriately to the method on the component implementing the service requested. The following sections describe how the XMC SOAP Engine 30 performs these tasks to support SOAP requests.

Please amend the third and fourth paragraphs on page 35 as follows:

The ~~exemplary~~example XMC SOAP Engine 30 comprises several objects. These objects work together to perform each requested SOAP operation. In addition, the XMC SOAP Engine 30 uses the XMC Motion Server 50 to eventually carry out the actual service request, either directly or using the ROPE module 56.

The ~~exemplary~~example XMC SOAP Engine 30 is a standard extension module for the Microsoft Internet Information module 74. As such, the XMC SOAP Engine 30 exposes the GetExtensionVersion, HttpExtensionProc, and TerminateExtension functions. These functions are called by module 74 on each request.